CSCE3301 – Computer Architecture

**Summer 2022**

**Project 1: RISC-V processor**

**Project Report**

Project Overview:

RISC-V processor simulated using Xilinx Vivado or any other Verilog simulator

Implementation language:

Verilog

A Brief Description of Code Implementation

The RISC-V processor is implemented using Verilog and simulated using Vivado. In this milestone, the processor is implemented as a single-cycle processor. Firstly, we started creating all the modules included in our processor datapath. We created modules for ALUcontrol , ControlUnit , Data Memory , DFilpFlop , 32-bit Multiplexer 2x1 , 32-bit Multiplexer 4x1 , Branching Unit , Shifting Unit,Instruction Memory , Program Counter, Register File ,Full Adder and Ripple Carry adder. The Immediate generator ,ALU and DEFINES were provided module code as support project codes.Then finally we joined all these modules in our top module “FullDataPath”. The following sections will discuss and describe each module's implementation and function.

ALUcontrol Module

The ALUcontrol module consists of 3 inputs and 1 output. For the inputs we have 32-bit instruction , 2-bits ALUOP, and 1-bit immtype . We implemented a wire named “inst1” to take the func3 of the instruction, which is 3 bits from bit 12 to bit 14, and another wire named "inst2” which is assigned to be the 30th bit of the instruction to be used for instructions that have the same opcode and funct3 but differ in their 30th bit , for instance the instructions ADD and SUB . Then we created an always block that consists of if statements basing the conditions on the 2-bits ALUOp input , “inst1” , “inst2” and immtype . In the always block, the output is assigned to be the instruction defined in the”DEFINES” file provided according to each instruction’s opcode , funct3(inst1), 30th bit (inst2) its immtype .

ControlUnit Module

In this module,we implemented the control unit by having the 32-bit instruction as an input . For the output , we created 10 signal outputs: memRead,memtoReg,memWrite , ALUSrc , Regwrite ,jal,immtype , 2-bits each, ALUOp, MUXSel, and Branch. In the module we created an always block that consists of a case statement on a wire called temp that is assigned to 5 bits from bit 2 to bit 6 of the 32-bit instruction input . Then we assign our 40 instruction opcodes of 9 bits to be the signals mentioned above . For instance (OPCODE\_Arith\_R : outputss = 9'b100010001;) here we assigned the opcode of all the R-format instructions to be 9-bits (100010001). Then finally, we assigned all the output signals to be equal to each bit accordingly to "outputss.”

Data Memory

The data memory module is designed to be byte-addressable. Therefore, for example, we instantiated the memory of X0 as ({mem[3], mem[2], mem[1], mem[0]}=17;//x0). This module consists of 2 always blocks, one based on the signal memWrite and the other based



Only two pages were converted.

Please **Sign Up** to convert the full document.

[**www.freepdfconvert.com/membership**](https://www.freepdfconvert.com/membership)